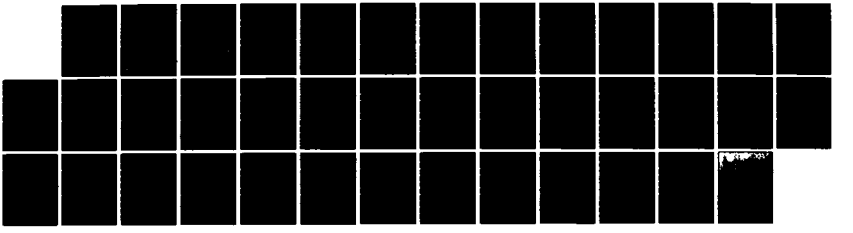
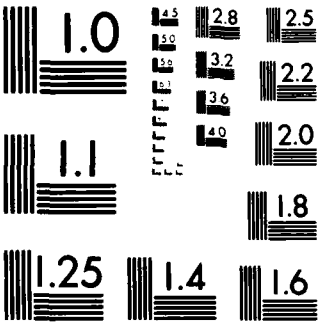
AD-A130 401 POSSIBLE APPROACHES FOR INTERFACING POD TO PSL AND SEM 1/1
(U) BGS SYSTEMS INC WALTHAM MA J P BUZEN ET AL. SEP 81
RPT-POD-0981-03 N00039-81-C-0183

UNCLASSIFIED F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963 A

Possible Approaches for
Interfacing POD to PSL and SEM

Contract N00039-81-C-0183

DTIC FILE COPY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1 REPORT NUMBER	2 GOVT ACCESSION NO
3 RECIPIENT'S CATALOG NUMBER	
4 TITLE (and Subtitle)	5 TYPE OF REPORT & PERIOD COVERED
Possible Approaches for Interfacing POD to PSL and SEM	
6 PERFORMING ORG. REPORT NUMBER	RPT-POD-0981-03
7 AUTHOR(s)	8 CONTRACT OR GRANT NUMBER(s)
Jeffrey P. Buzen, Glynn Glacone, Donald E. Hal Peter S. Mager, Richard T. Williams	N-00039-81-C-0183
9 PERFORMING ORGANIZATION NAME AND ADDRESS	10 PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
BGS Systems, Inc. WALTHAM MA	ACRN: Item 0001
11 CONTROLLING OFFICE NAME AND ADDRESS	12 REPORT DATE
Naval Electronic Systems Command Washington, DC 20360	September, 1981
	13 NUMBER OF PAGES 33
14 MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15 SECURITY CLASS. (of this report) UNCLASSIFIED
	15a DECLASSIFICATION/DOWNGRADING SCHEDULE

16 DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release and sale; its distribution is unlimited.

17 DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18 SUPPLEMENTARY NOTES

DTIC ELECTE JUL 15 1983 A

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)

POD, PSL/PSA, Performance, Modeling, Programs Specification Language, Program Design, Performance Oriented Design Requirements Specification

20 ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the most promising strategies for building a "bridge" type interface between POD and either PSL/PSA or SEM. In addition a cookbook type recipe is given for constructing a POD SDF from PSA output reports derived from PSL system descriptions.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

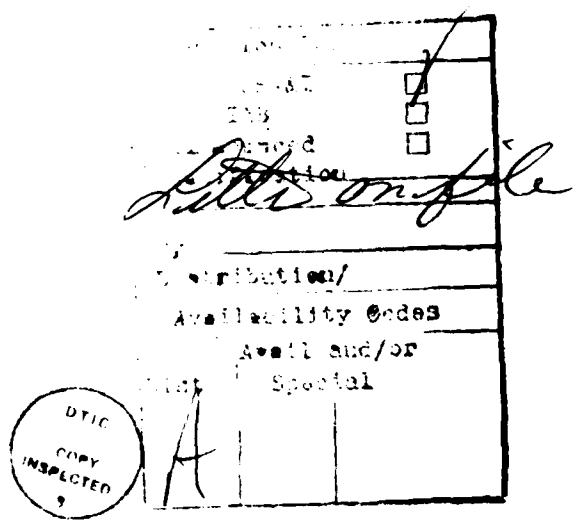
UNCLASSIFIED

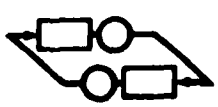
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

29 September 1981

Possible Approaches for Interfacing POD to PSL and SEM

Jeffrey P. Buzen
Glynn Giacone
Donald E. Hall
Peter S. Mager
Richard T. Williams
RPT-POD-0981-03
27 September 1981





BGS Systems, Inc.
P.O. Box 128
Lincoln, MA 01773
(617)-890-0000

Table of Contents

1	Introduction	2
2	PSL/PSA and SEM	2
3	Architecture of a SEM - POD Interface	6
4	BGS Options for Interfacing PSL to POD	6
5	Conclusion	9
6	PSA Reports Useful for Manually Producing an SDF	11
7	Cookbook Approach to Building a POD Model from PSA Reports ...	20

1 Introduction

This report summarizes the options available to BGS Systems with respect to building an interface between the PSL facility (developed by the ISDOS group at the University of Michigan) and POD. The choices span a range in sophistication, user appeal, and amount of effort involved. The appropriate choice should probably be based on the extent to which the more sophisticated options could be expected to make POD and PSL/PSA easier to use together and on the demands of the POD user community. An appendix describes a manual procedure for converting PSL system descriptions to a POD SDF and describes PSA reports that are useful aids for performing this translation.

2 PSL/PSA and SEM

An important issue is that there are really two ISDOS products that we can interface to:

* PSL/PSA is the older product with an established user base (including some current and potential POD users at NSWC, NUSC, NOSC, and NAVAIR).

* SEM is a newer facility with very few users, but greater flexibility and a good deal of future potential.

An overview of how these systems function is shown in Figures 1 (PSL/PSA) and 2 (SEM).

PSL/PSA consists of a Problem Statement Language (PSL) and a Problem Statement Analyzer (PSA). A description of a system, written in PSL, is fed into PSA, which then produces a number of reports describing the system. The reports to be produced are specified using PSA commands and directives. This is illustrated in figure 1.

1 Introduction

This report summarizes the options available to BGS with respect to building an interface between the PSL facility (developed by the ISDOS group at the University of Michigan) and POD. The choices span a range in sophistication, marketing appeal, and amount of effort involved. The appropriate choice should probably be based on the extent to which the more sophisticated options could be expected to make POD and PSL/PSA easier to use together and on the demands of the POD user community. An appendix describes a manual procedure for converting PSL system descriptions to a POD SDF and describes PSA reports that are useful aids for performing this translation.

2 PSL/PSA and SEM

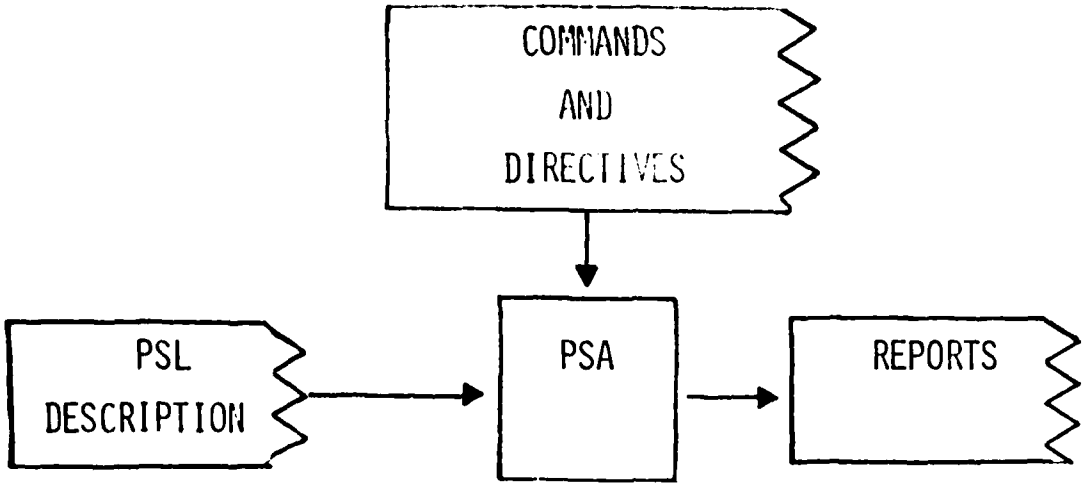
An important issue is that there are really two ISDOS products that we can interface to:

* PSL/PSA is the older product with an established user base (including some current and potential POD users at NSWC, NUSC, NOSC, and NAVAIR).

* SEM is a newer facility with very few users, but greater flexibility and a good deal of future potential.

An overview of how these systems function is shown in Figures 1 (PSL/PSA) and 2 (SEM).

PSL/PSA consists of a Problem Statement Language (PSL) and a Problem Statement Analyzer (PSA). A description of a system, written in PSL, is fed into PSA, which then produces a number of reports describing the system. The reports to be produced are specified using PSA commands and directives. This is illustrated in figure 1.



USED FOR:

* DOCUMENTATION OF INTERFACES

* DETAILED DESCRIPTION OF RELATIONSHIPS AMONG DATA ITEMS AND BETWEEN DATA AND PROGRAMS

* CAN CHECK CONSISTENCY AND COMPLETENESS OF SYSTEM SPECIFICATION

FIGURE 1: Problem Specification Language/Problem Statement Analyzer (PSL/PSA)

The System Encyclopedia Manager (SEM) is a more elaborate facility that includes three subsystems - a Database Manager, the Generalized Analyzer (GA) and a Query Facility - as well as a separate front end, META, for describing system definition languages. A language, such as PSL or POD, is described to META in its metalanguage. META processes this language definition and produces

* an internal database used by SEM
* a language reference manual
* various diagnostic checks and reports about the language.

A user enters a system description to SEM in the appropriate description language, in this case either PSL or a POD SDF. SEM references its language definition table database to interpret this description and enter it into a database containing information about the system. A database management subsystem is used as the interface to the database of system descriptions. A report facility, the Generalized Analyzer, and an interactive Query Facility are available to analyze the database and produce reports and validation checks on the system. This is illustrated in Figure 2.

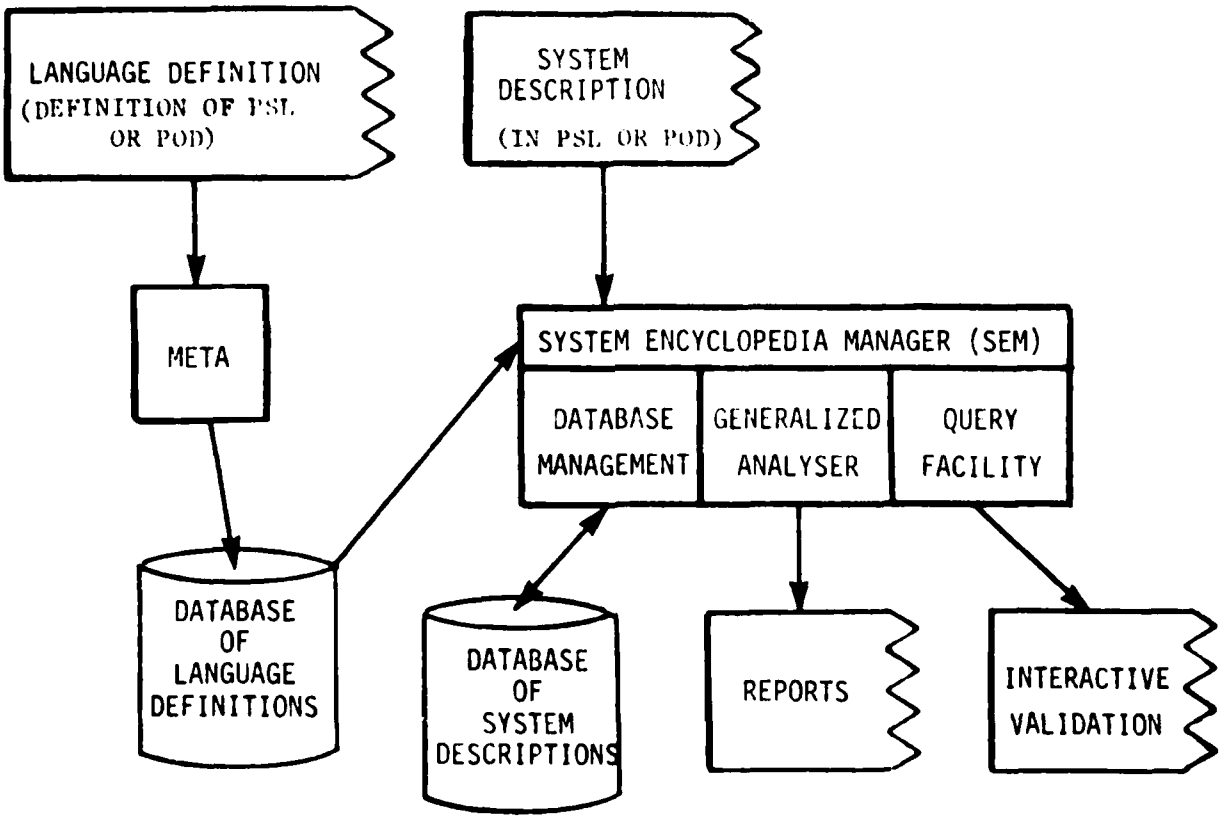


FIGURE 2: The System Encyclopedia Manager (SEM)

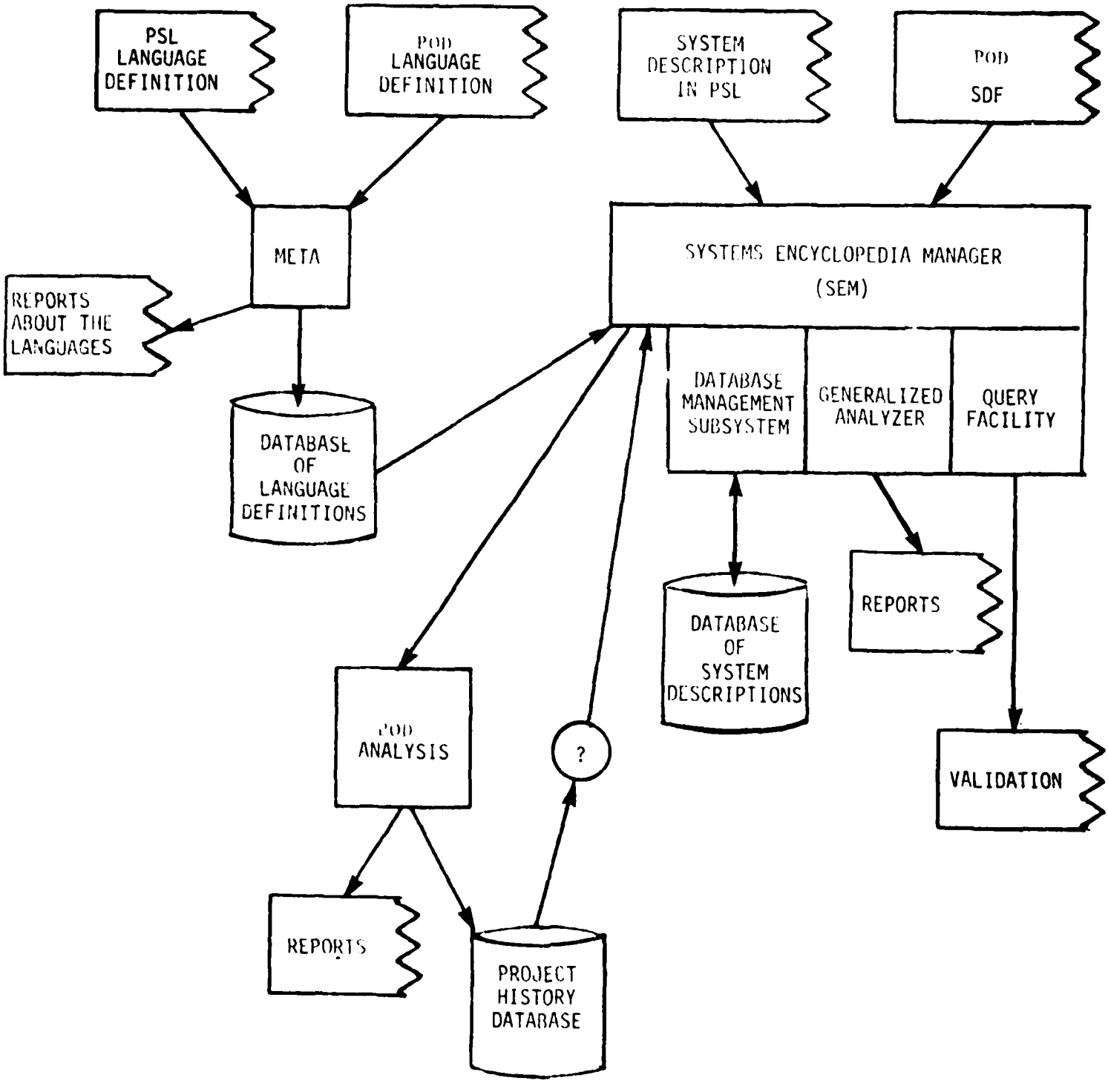


FIGURE 3: Architecture of a SEM-POD Interface

SEM is more general than PSL/PSA and, thus, easier to interface to other systems, but this generality reduces the ability of the Generalized Analyzer to generate specific reports - so certain PSA reports are not currently available under SEM.

3 Architecture of a SEM - POD Interface

An overview of how POD and PSL would interact using SEM is shown in Figure 3. In the figure definitions of PSL and POD are fed into META, which produces various reports about each of them and enters the language definitions into the language definition database. A PSL description of a system and a POD SDF are input to SEM. SEM uses the language definition database to interpret the system descriptions and enters the processed system descriptions into its system description database. Reports based on these descriptions can then be produced using the Generalyzed Analyzer and validation checks can be done with the Query Facility. The POD SDF can be input to POD to analyze projected system performance and the results fed back to the database using the Project History Database (PHDB) facility and a SEM interface yet to be specified. This is perhaps the cleanest and most flexible approach to interfacing the two systems.

4 BGS Options for Interfacing PSL to POD

Three alternative approaches that could interface PSL/PSA and POD without going through SEM are illustrated in Figure 4. The first and simplest approach (option 1) involves taking PSA output reports and using the information contained in them to manually build a POD SDF. The main PSA reports that could be useful for doing this are summarized and illustrated in Appendix A.

Automatically generating a POD SDF from PSA output reports would not be feasible since the precise formats are volatile (subject to change) and contain large numbers of extraneous characters included for aesthetics and formatting rather than informational content. In addition, the information needed by POD is divided among PSA reports and merging these into a single SDF could be cumbersome. In addition, the operational procedures would be

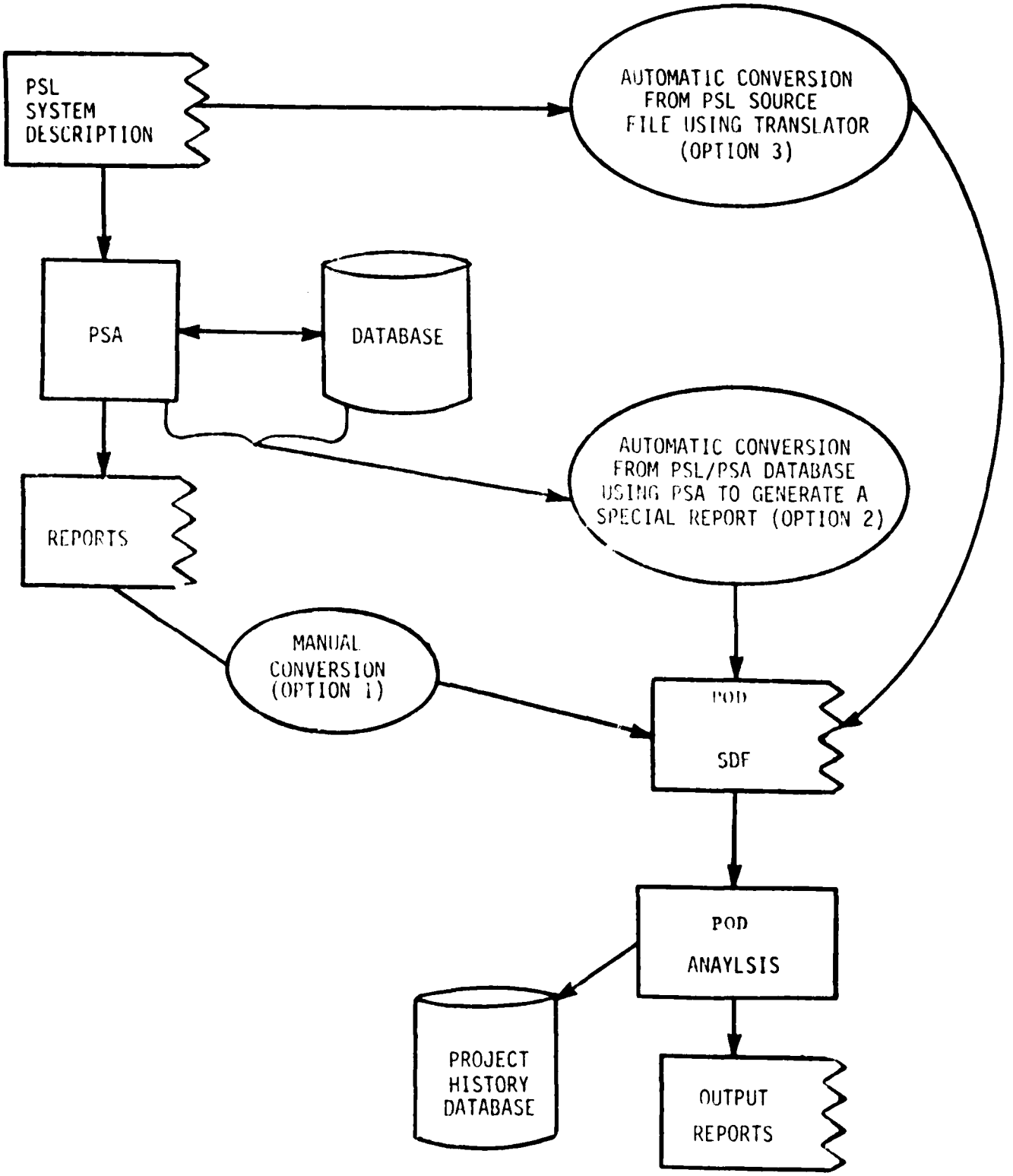


FIGURE 4: BGS Options for Interfacing PSL to POD

more complex than in options 2 and 3.

Option 2 involves automatically generating a POD SDF from information contained in the PSA system description database. This is a promising approach but would involve getting access to the PSA internal formats and interfaces and would thus involve a joint effort by BGS Systems and ISDOS. In this case a POD SDF file would be generated as a special PSA report. This approach would have a natural extension to a SEM interface if the SDF report was generated using the Generalized Analyzer instead of PSA. ISDOS seems used to making arrangements (providing the necessary support) for the generation of such reports (which it calls "bridges") in the SEM environment.

Option 3 involves directly translating a PSL system description into a POD SDF. This involves building a translator that was independent of PSA and the Generalized Analyzer. If a SEM interface was used, its Query Facility could be used to compare the PSA and SDF description and check for consistency and completeness. This option is probably less desirable than option 2 since system descriptions can sometimes be entered piecemeal so that no single PSL description would contain a complete and current view of the system. By contrast, the PSA database should always contain the complete current specification for the system.

Both options 2 and 3 would benefit substantially from an extension of PSL that would encourage the user to provide more performance related information and to enter this data in a PSL description in a standardized way. Some of the issues involved in doing this are discussed in [1]. Under options 2 and 3, the resulting SDF would be more complete and would require less manual manipulation, tuning, and filling in of missing data. Even under option 1, users would become more conscious of performance related factors at an earlier stage of system specification. This would both facilitate the manual creation of an SDF and probably lead them to the design of better performing systems.

Finally we come to the issue of feedback of the results of POD analysis into the PSL reporting facilities. This could probably be done most easily via SEM using the POD Project History Data Base (PHDB) facilities. Formats would be defined to SEM using META. The PHDB data would be entered to the system description by using SEM. The generalized analyzer and query facility could then be used to report on POD results in a way analogous to how SAS is being used currently. It is likely, however, that we would choose to define more highly structured reports and data formats to be used with this type of system than we are currently doing using SAS. An alternative approach that should also be considered is to have POD generate PSL specifications of performance related quantities as a special report

option. One possible implementation of this would be to use the PHDB external interface to write out the appropriate performance related information to an external file which would then be processed by a report facility to generate a PSL description.

5 Conclusion

In summary, we have described the main options available for interfacing POD and PSL/PSA/SEM. A direct PSL - SDF translator would be an appropriate tool to implement if only limited ISDOS support can be expected and a PSL interface is expected to get substantial use. A new PSA or SEM/GA report from the system description database would be a preferable alternative if substantial ISDOS cooperation is available or a joint effort is undertaken. This would also provide the most robust interface since the database system description is generally more complete and current than any single PSL description. A manual conversion would be the best approach if only limited joint use is expected and/or BGS wishes to limit the amount of resources it devotes to this project. In any case, extension of PSL to include more performance oriented information using POD terminology would be beneficial

* by making users more aware of performance at an earlier stage in system specification
* by getting them used to the POD operational terminology and way of looking at performance
* and by facilitating whatever PSL to POD conversion (manual or automatic) might be undertaken.

Appendix A

6 PSA Reports Useful for Manually Producing an SDF

The most useful PSA reports that could serve as aids in producing a POD SDF file are:

* the structure report
* the picture report
* the process chain report
* the frequency report
* the resource consumption report
* system size and volume reports

These reports are described and illustrated in this appendix.

The PSA Structure Report gives a breakdown of the module calling hierarchy of a system. This is useful in producing a skeleton of the module specification section of the SDF. A typical structure report is shown in Figure A-1.

The PSA Picture Report shows the interfaces of a system including the processes and events that trigger and invoke a particular process (control interface) and the inputs used and outputs generated by the process. This gives a visual picture of the data and control flow through a system and the files that may be used by a module. A typical picture report is shown in Figure A-2.

The PSA Process Chain Report shows the control flow through a system in terms of which events/processes trigger or call which other events and processes. It summarizes the control flow through a system. A Process chain report is shown in Figure A-3.

The PSA Frequency Report relates system inputs and process invocation to the frequency with which they occur. This frequency may be dependent on a system parameter. This report is useful in determining workload arrival rate, file access, and number of times a module call will be invoked and

how many iterations will be made through a loop. A Frequency report is shown in Figure A-4.

The Resource Consumption Analysis Report is available with release 5.1. of PSL/PSA and summarizes specific resource utilization information. Note that the information is not available in system descriptions based on earlier releases of PSL/PSA (including the one in use at NSWC). The report contains the information needed to generate the server usage statements in the SDF module specification sections and, where available, is among the most useful PSA reports. Because of its potential usefulness, a more detailed description is given below.

The Resource Consumption Analysis report of PSA presents processors' consumption of resources in performing the specified process and its component processes. The analyzer performs a walk of the SUBPARTS_UTILIZES network which starts at the specified PROCESS name and extends to the specified number of levels. The resource consumption and frequency information is computed for each visit to each process in the structure. Since a process may be visited many times, its resource consumption may be computed many times as well. Depending upon the viewpoints, two formats - BYPROCESSOR and BYRESOURCE - may be used in PSA to provide resource consumption status about each PROCESS. This is illustrated by the following example:

ROOT PROCESS: UPDATE_RECORD

PROCESSOR CENTRAL_PROCESSOR IS INVOKED 800 TIMES PER HOUR

RESOURCE CONSUMED	AMOUNT CONSUMED	MEASURED IN
CPU_TIME	215.5	MSEC

PROCESSOR DISI_A IS INVOKED 800 TIMES PER HOUR

RESOURCE CONSUMED	AMOUNT CONSUMED	MEASURED IN
DISK_A_TIME	103.0	MSEC

The above report is presented in BYPROCESSOR format.

If BYRESOURCE format is used, the report may appear as below:

```
ROOT PROCESS:   UPDATE_RECORD

RESOURCE  CPU-TIME  MEASURED IN MSEC

PROCESSOR CONSUMING     FREQUENCY     AMOUNT CONSUMED
   CENTRAL_PROCESSOR     800/HOUR           215.5

RESOURCE DISK_A_TIME   MEASURED IN MSEC

PROCESSOR CONSUMING     FREQUENCY       AMOUNT CONSUMED
    DISK_A            800/HOUR 103.0
```

System Size and Volume Reports summarize the information related to size and volume that is usually entered as system parameters to PSL/PSA. These reports are only available from release 5.1 and later releases of PSL/PSA.

PSA Version A4.2[illegible]

PSL/PSA Version 4.2 Example

Structure Report

COUNT	LEVEL	NAME
1	1	payroll-processing
2	2	new-employee-processing
3	3	salaried-information-creation
4	3	hourly-information-creation
5	3	hire-report-entry-generation
6	3	department-file-addition
7	2	terminating-emp-processing
8	3	salaried-information-deletion
9	3	hourly-information-deletion
10	3	term-report-entry-generation
11	3	department-file-removal
12	2	employee-processing
13	3	hourly-employee-processing
14	4	hourly-paycheck-validation
15	5	time-card-validation
16	4	hourly-emp-update
17	5	hours-update
18	4	h-report-entry-generation
19	4	hourly-paycheck-production
20	5	h-gross-pay-computation
21	5	total-hours-computation
22	3	salaried-employee-processing
23	4	salaried-paycheck-validation
24	4	salaried-emp-update
25	4	s-report-entry-generation
26	4	salaried-paycheck-production
27	5	s-gross-pay-computation
28	2	process-library
29	3	pay-computation-validation
30	3	tax-computation
31	3	net-pay-computation
32	3	total-deductions-computation
33	3	gross-pay-update
34	3	federal-deductions-update
35	3	state-deductions-update

FIGURE A-1

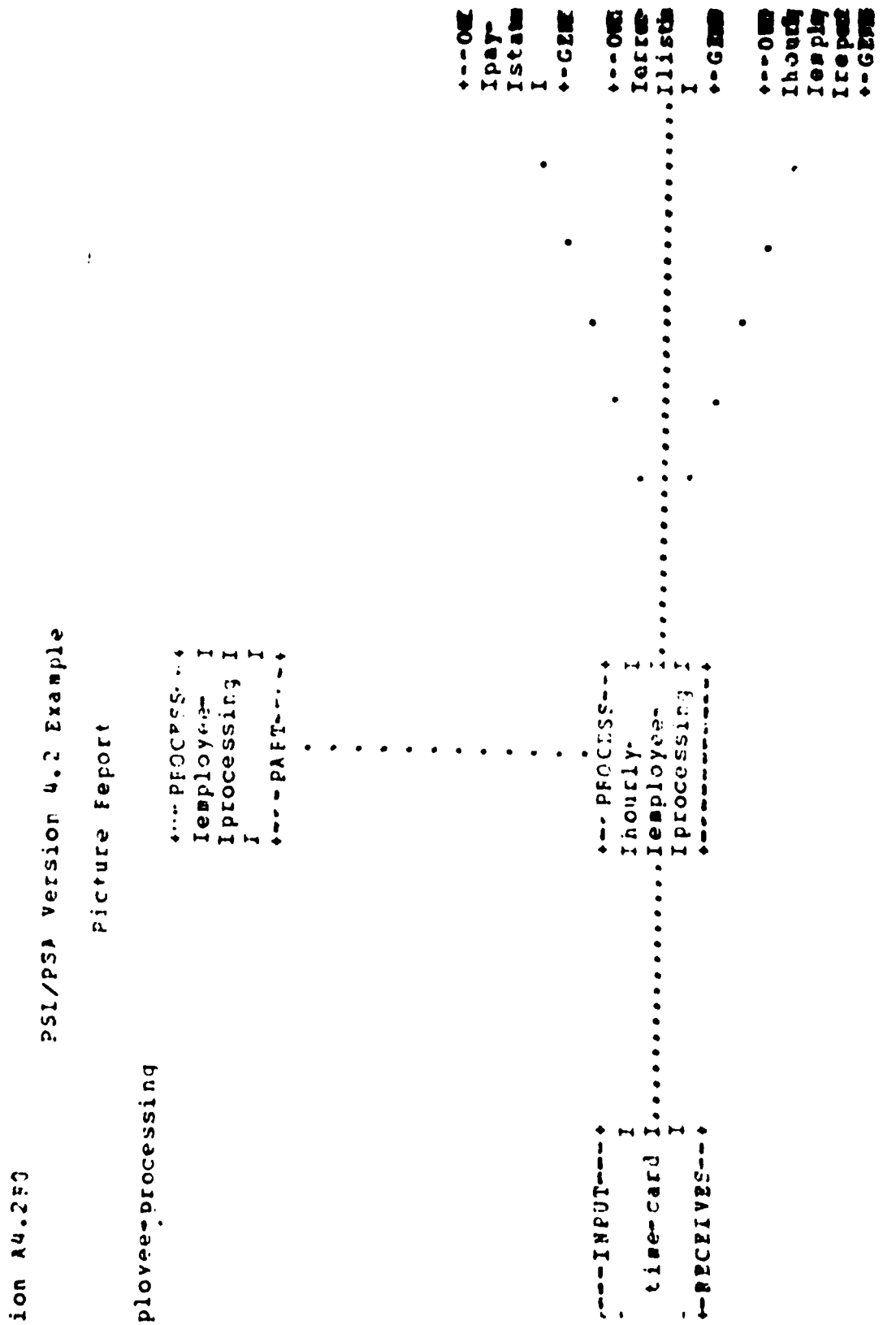

PSA Version A4.250
hourly-employee-processing
PSL/PSA Version 4.2 Example
Picture Report
INPUT
time-card
RECEIVES
PROCESS
hourly
employee
processing
PART
PROCESS
employee
processing

FIGURE A-2

PSA Version A4.2[illegible]

PSL/PSA Version 4.2 Example

Process Chain

INITIAL NAME = salaried-emp-processing-init

```
                                                                                  +---EVENT---+
                                                                                  Ihourly-    I
                                                                                  Idata-      I
                                                                                . Icollection I
                                          +---EVENT---+        +--PROCESS--+   .  +ON INCEPTN-+
                                          Ihourly-emp-I        Ihourly-    I .
                                       ...Iprocessing-I.......Iemployee-  I
                                       .  Iinit       I        Iprocessing I.
                                       .  +ON INCEPTN-+        +-TRIGGERED-+ .
+---EVENT---+        +--PROCESS--+ .                                        .  +---EVENT---+
Isalaried-e-I        Isalaried-  I.                                          .  Inew-employ-I
Imp-process-I.......Iemployee-  I                                          ...Iee-process-I
Iing-init   I        Iprocessing I.                                             Iing-init   I
+-----------+        +-TRIGGERED-+ .                                             +--ON TERM--+
                                       .
                                       .  +---EVENT---+        +--PROCESS--+      +---EVENT---+
                                       .  Isalaried-  I        Isalaried-  I      Isalaried-  I
                                       ..Idata-      I.......Ipaycheck-  I.......Ipaycheck-  I
                                          Icollection I        Ivalidation I      Iprod-init  I
                                          +ON INCEPTN-+        +-TRIGGERED-+      +--ON TERM--+
```

FIGURE A-3

PSL/PSA Version 4.2 Example

Frequency Report

INTERVAL: month

APPLIES TO	TYPE	FREQUENCY
passed-error-checks	EVENT	valid-check-rate
salaried-emp-processing-init	EVENT	no-salaried-emp-processing
validity-check	EVENT	error-rate-per-month
employment-termination-form	INPUT	several
hourly-employment-form	INPUT	several
salaried-employment-form	INPUT	several
tax-withholding-certificate	INPUT	several
error-listing	OUTPUT	no-of-payroll-processing
salaried-employee-report	OUTPUT	no-salaried-emp-processing
hourly-employee-processing	PROCESS	at-least-one
hourly-paycheck-production	PROCESS	at-least-one
pay-computation-validation	PROCESS	error-rate-per-month
payroll-processing	PROCESS	no-of-payroll-processing
salaried-employee-processing	PROCESS	no-salaried-emp-processing
salaried-paycheck-production	PROCESS	no-salaried-emp-processing
time-card-validation	PROCESS	error-rate-per-month

INTERVAL: week

APPLIES TO	TYPE	FREQUENCY
hourly-emp-processing-init	EVENT	no-hourly-emp-processing
new-employee-processing-init	EVENT	no-new-emp-processing
termination-processing-init	EVENT	no-terminating-emp-processing
time-card-missing	EVENT	time-card-error-rate
validity-check	EVENT	error-rate-per-week
time-card	INPUT	no-of-hourly-employees
hourly-employee-report	OUTPUT	no-hourly-emp-processing
terminated-employee-report	OUTPUT	no-terminating-emp-processing
hourly-employee-processing	PROCESS	no-hourly-emp-processing
hourly-paycheck-production	PROCESS	no-hourly-emp-processing
new-employee-processing	PROCESS	no-new-emp-processing

FIGURE A-4

Appendix B

7 Cookbook Approach to Building a POD Model from PSA Reports

In the interval until automated tools become available, PSL/PSA users who wish to build a POD model can make use of a number of PSA reports that provide relevant information about the system to be modeled. In this way, PSL/PSA users can capture the information they have already collected about their system in a PSA database and incorporate that information into the POD model. As an aid to doing this the following six step procedure is suggested.

1. Identify modules and calling relationships using the PSL structure report.

2. Identify file usage and other I/O using the PSL picture report.

3. Identify workloads and their relationship to modules using the PSL process chain report.

4. Incorporate looping and case distribution using the PSL process chain report.

5. Determine workload, looping, and file access rates using the PSL frequency report.

6. Incorporate actual resource demands using the PSL resource consumption analysis report.

The diagrams on the following pages show how the PSL/PSA to POD model transformation can be made using a very simple system as an example.

1. The PSL structure report for processes shows the breakdown of processes (modules) by the subprocesses (modules) that it calls. This hierarchical view of system structure can be translated into POD calling relationships. This is illustrated in Figure 1.

2. The PSA Picture Report links processes (modules) to the inputs and outputs they interface with. This information can be used to determine system file access patterns. This is shown in Figure 2.

3. The process chain report is then used to associate processes

Figure 1: Identifying the Tree Structure Calling Hierarchy

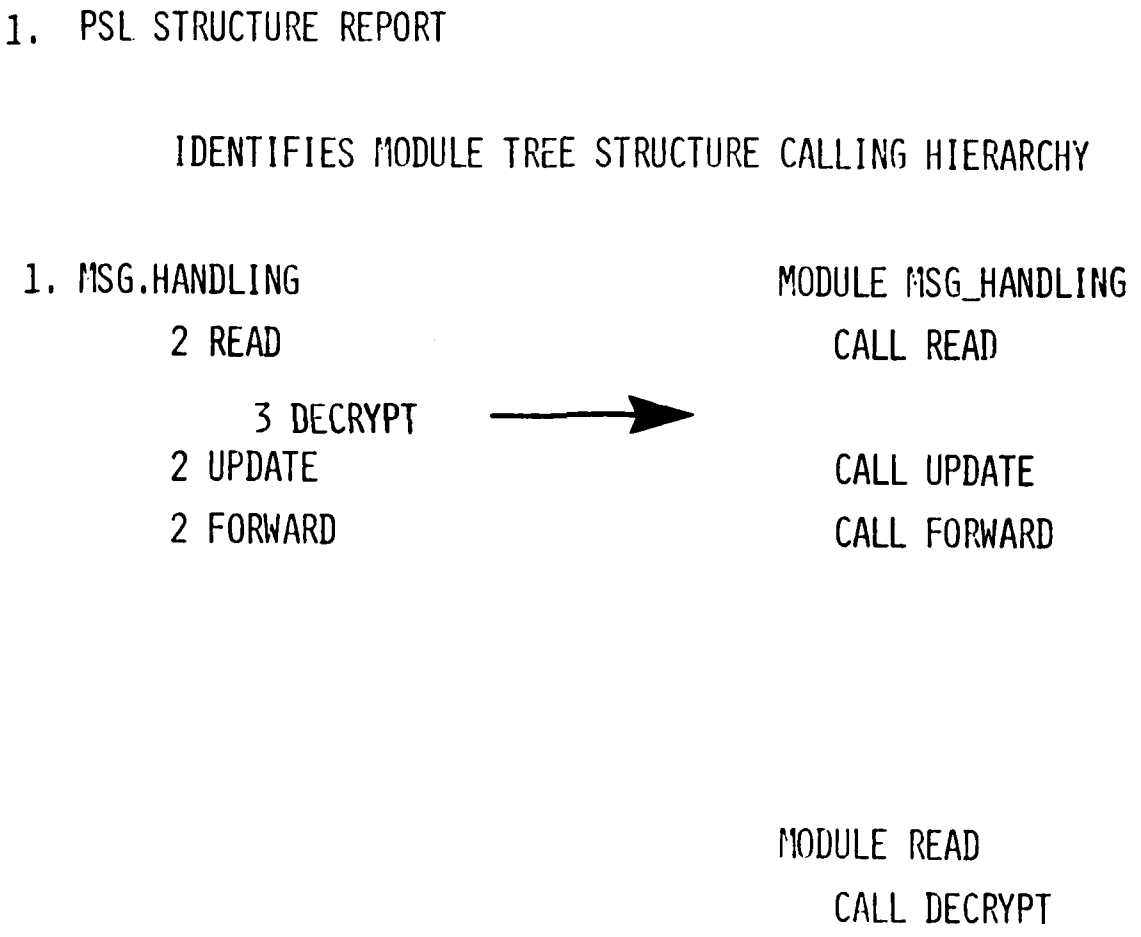
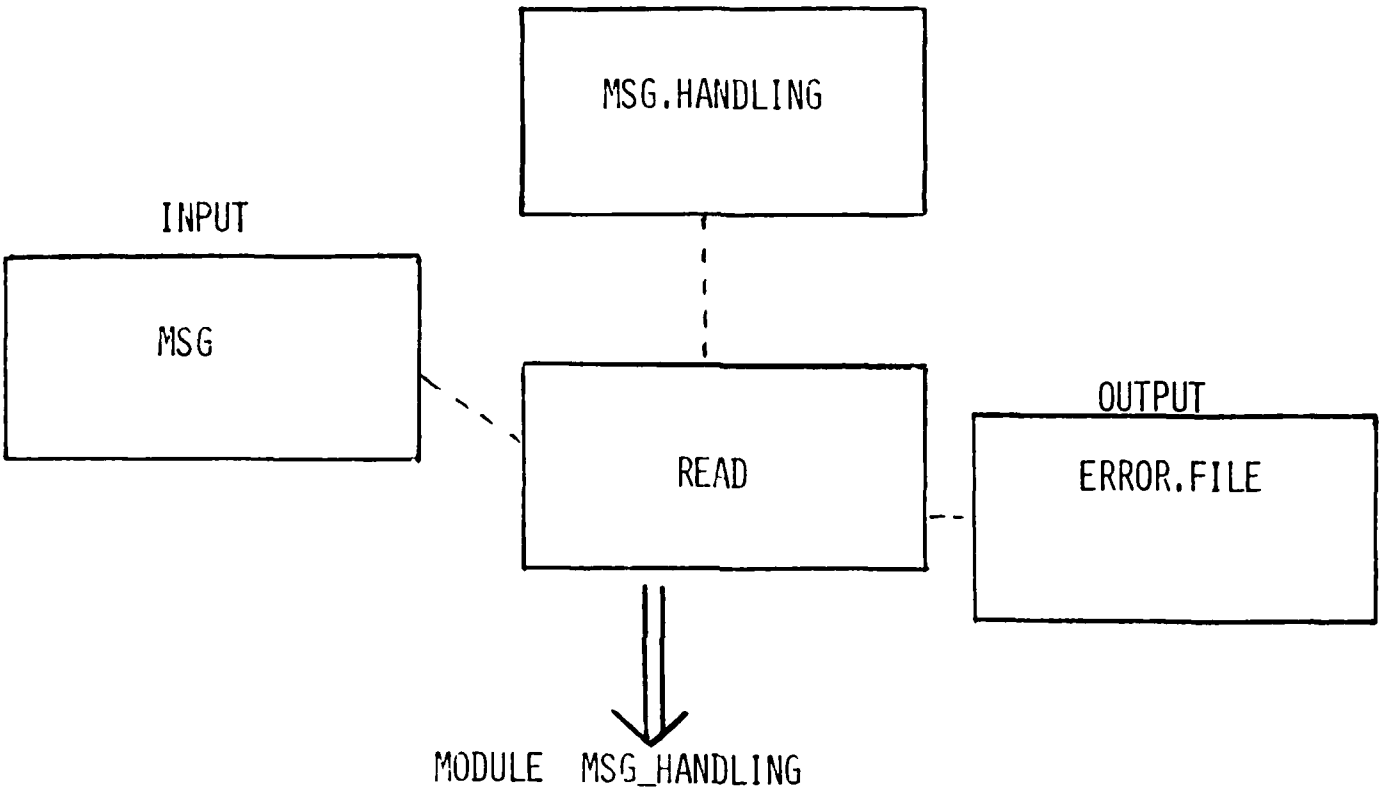


Figure 2: Identifying File Reads and Writes




```
MODULE READ

                    EST   MSG USAGE = N READS
                    EST   ERROR_FILE USAGE = M WRITES

END
```

(modules) with the events (workloads) that cause them to be invoked. This is illustrated in Figure 3.

4. The Process Chain Report also contains information that can be used to introduce loop and case structures into the POD model. Case structures are introduced when a process (module) invokes other processes (calls submodules or invokes other processing subsequences) with probabilities dependent on some external condition(s) being true. Looping is introduced when the frequency of invocation of some processing subsequence is dependent on a condition or parameter. This is illustrated in Figure 4.

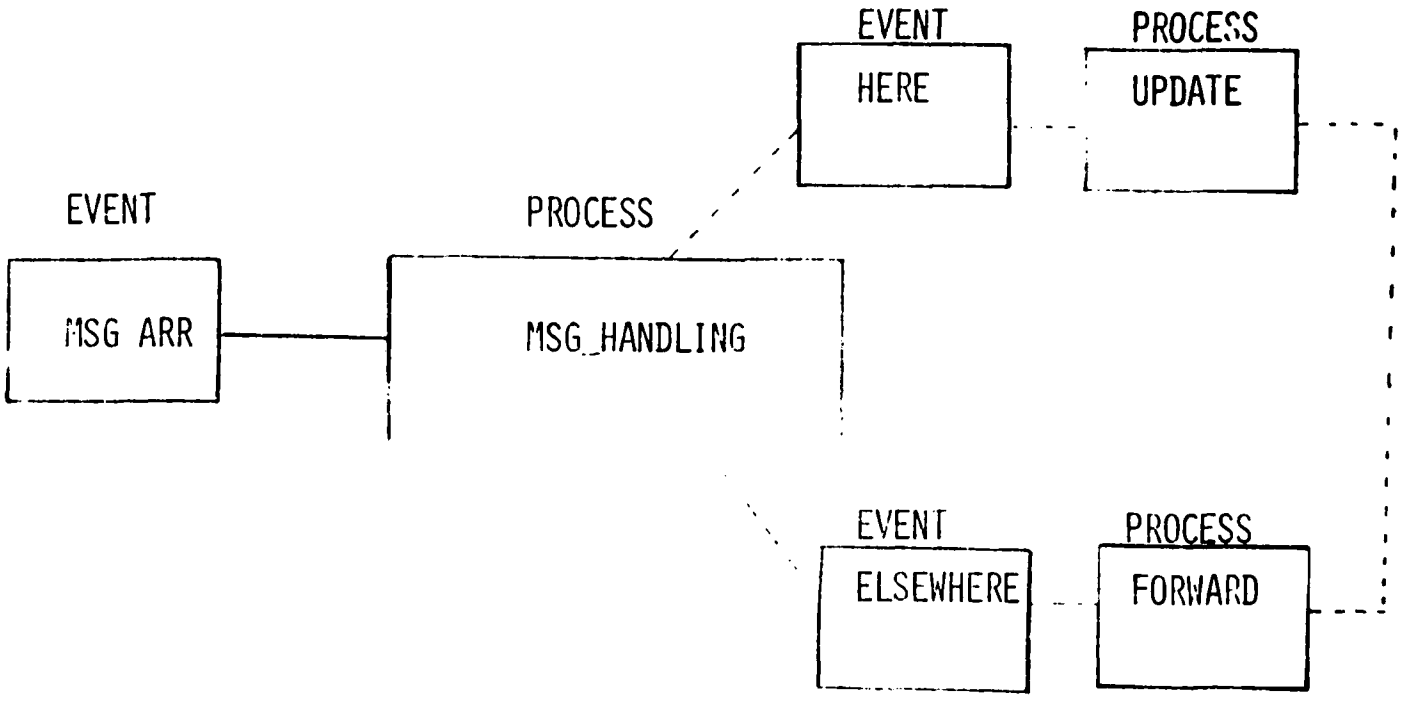
5. The Frequency Report is used to associate actual values or formal parameters to the dummy parameters previously filled in from information in the Process Chain and Picture Reports. In this way the workload arrival rate, degree of looping, and number of file accesses can be filled in. This is illustrated in Figure 5.

6. Finally the actual resource consumption estimates for the POD model are filled in. This can be gotten from version 5.1 and later releases of PSL/PSA using the Resource Consumption Analysis Report. In earlier releases this information would have to be collected separately (or included in the PSL description as a comment or attribute value field and gotten from a report showing values of that attribute).

The state of a typical POD SDF after each stage of this process is illustrated Figures B6A-B6G.

Sample printouts of the PSA Structure, Picture, Process Chain and Frequency Reports are shown in Figures A1-A4.

Figure 3: Identifying Workloads and their Relationships to Modules



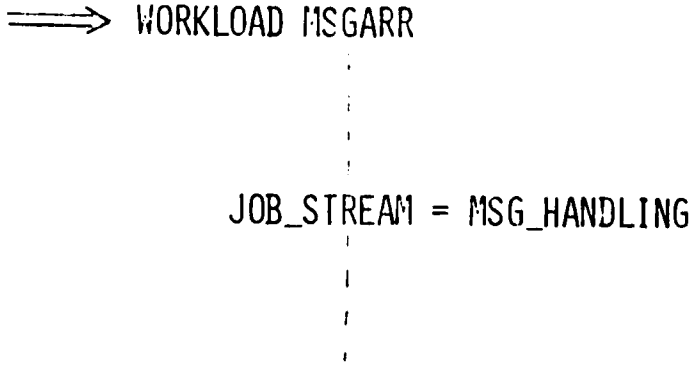
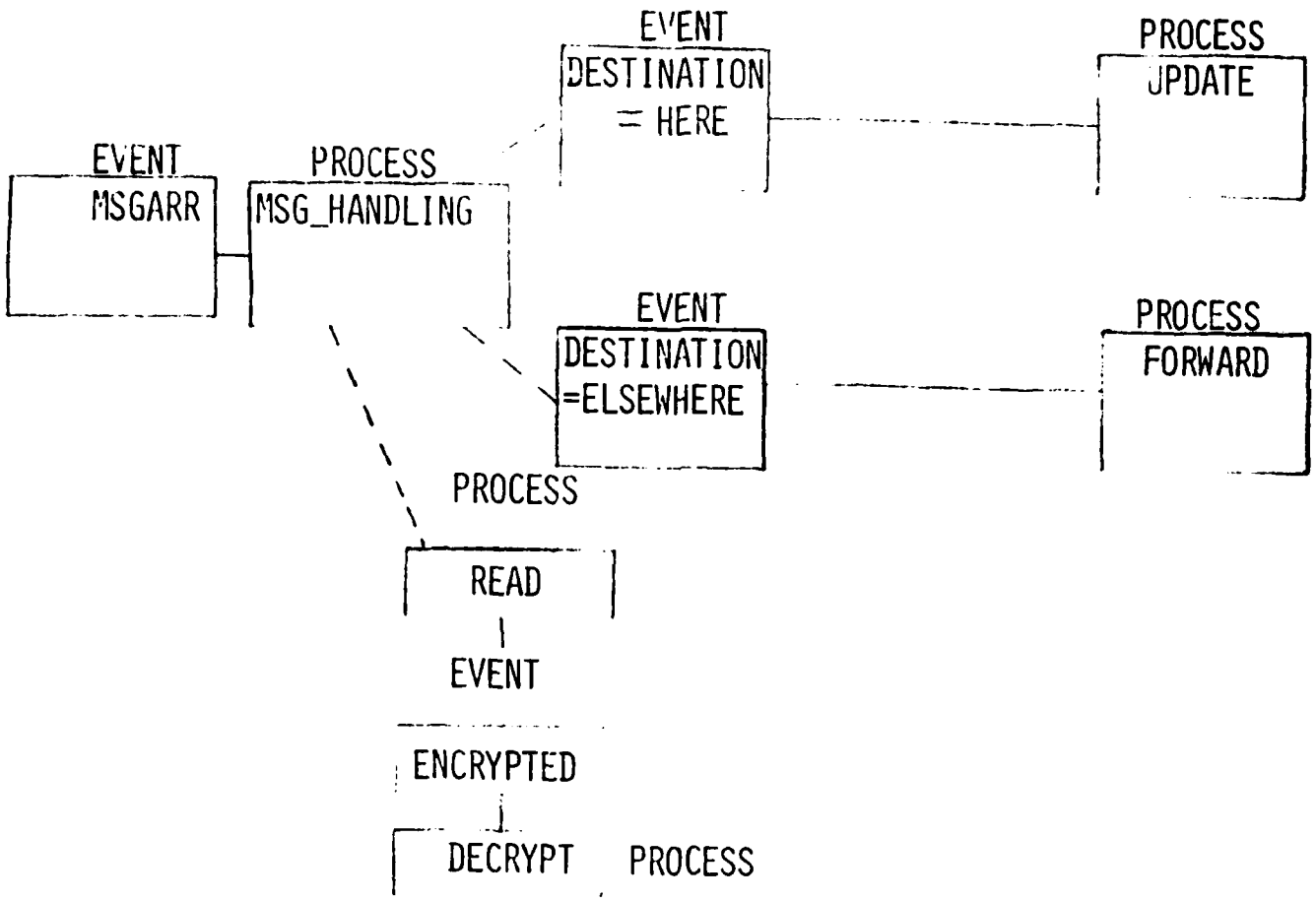


Figure 4: Incorporating Looping and Case Distribution




```
MODULE MSG_HANDLING
     CALL READ
     TEST DESTINATION
          CASE 'HERE'
               CALL UPDATE
          CASE 'ELSEWHERE'
               CALL FORWARD
     ENDTEST
END
```

```
MODULE READ

LOOP ENCRYPTED TIMES
    CALL DECRYPT
ENDLOOP
```

Figure 5: Determining Workload, Looping, and File Access Rates

```
INTERVAL: HOUR

     APPLIES TO              TYPE        FREQUENCY

     MSG ARR                 EVENT       1000

     ENCRYPTED               EVENT       PARM1*MSGARR

     MSG                     INPUT       1*MSGARR

     ERROR_FILE              OUTPUT      3*MSGARR

MODULE READ
     EST  MSG USAGE  = 1 READ
          LOOP PARM1 TIMES
               CALL DECRYPT
          ENDLOOP
     EST  ERROR_FILE USAGE = 3 WRITES
END

WORKLOAD MSGARR

     ARRIVAL_RATE = 1000 MESSAGES/HR
```

PROTOTYPICAL MODULE SPECIFICATION

AFTER STEP 1

```
MODULE MSG_HANDLING

     CALL READ

     CALL UPDATE

     CALL FORWARD

END

MODULE READ

     CALL DECRYPT

END
```

FIGURE B6-A

PROTOTYPICAL MODULE SPECIFICATION

AFTER STEP 2

```
MODULE MSG_HANDLING

     CALL READ

     CALL UPDATE

     CALL FORWARD

END

MODULE READ

|    EST MSG USAGE = N READS
       CALL DECRYPT
|    EST ERROR-FILE USAGE = M WRITES
END
```

FIGURE 36-B

PROTOTYPICAL WORKLOAD. SPECIFICATION

AFTER STEP 3

```
WORKLOAD MSGARR


JOB_STREAM = MSG_HANDLING

END
```

FIGURE B6-C

PROTOTYPICAL MODULE SPECIFICATION

AFTER STEP 4

```
MODULE MSG_HANDLING



        CALL READ

    TEST DESTINATION
        CASE 'HERE'
        CALL UPDATE

        CASE 'ELSEWHERE'
        CALL FORWARD
    ENDTEST

END

MODULE READ
    EST MSG USAGE = N READS
        LOOP (ENCRYPTED) TIMES
            CALL DECRYPT
        ENDLOOP
    EST ERROR_FILE USAGE = M WRITES
END
```

FIGURE B6-D

PROTOTYPICAL MODULE SPECIFICATION

AFTER STEP 5

```
MODULE MSG_HANDLING



        CALL READ

    TEST DESTINATION
        CASE 'HERE'
        CALL UPDATE

        CASE 'ELSEWHERE'
        CALL FORWARD
    ENDTEST

END

MODULE READ
    EST MSG USAGE = 1 READS
        LOOP   PARM1    TIMES
            CALL DECRYPT
        ENDLOOP
    EST ERROR_FILE USAGE = 3 WRITES
END
```

FIGURE B6-E

PROTOTYPICAL WORKLOAD. SPECIFICATION

AFTER STEP 5

```
WORKLOAD MSGARR

ARRIVAL_RATE = 1000  /HR

JOB_STREAM = MSG_HANDLING

END
```

FIGURE B6-F

PROTOTYPICAL MODULE SPECIFICATION

AFTER STEP 6

```
MODULE MSG_HANDLING

1.  EST CPU1 USAGE = 300 MSEC
          CALL READ

     TEST DESTINATION
          CASE 'HERE'
          CALL UPDATE

          CASE 'ELSEWHERE'
          CALL FORWARD
     ENDTEST

END

MODULE READ
     EST MSG USAGE = 1 READ
          LOOP   PARM1    TIMES
               CALL DECRYPT
          ENDLOOP
     EST ERROR_FILE USAGE = 3 WRITES
END
```

FIGURE B6-G

REFERENCES

[1] Bunen, J.J., Goldberg, R.F., Hall, L.E., Hua, K., Langer, A.E., Levy, A.I., Meyer, P.F. and Schwenk, H.J.
On the Feasibility of Interfacing PSL and PSL/PSA.
Technical Report, BGN Systems, Inc., Waltham, Mass., December, 1980.

END

FILMED

8-83

DTIC